# LỜI NÓI ĐẦU

Chương trình dịch là một môn học của ngành khoa học máy tính. Cùng với việc phát triển của các chuyên ngành: ngôn ngữ hình thức và otomat ,và các chuyên ngành khác, lý thuyết thiết kế biên dịch ngày càng một hoàn thiện hơn.

Ngày nay lý thuyết biên dịch đã được ứng dụng không những trong việc thiết kế trình biên dịch cho ngôn ngữ lập trình mà còn trong lĩnh vực trí tuệ nhân tạo-hiểu ngôn ngữ tự nhiên,dịch máy…

Đồ án môn học:Chương trình dịch là dịp tốt để sinh viên nắm vững lý thuyết và cách thiết kế trình biên dịch được tốt.

Chúng em đã thực hiện đồ án với đề tài 10:

**Phân tích cú pháp bằng phương pháp đệ qui không quay lui cho ngôn ngữ C**

Chúng em xin chân thành cảm ơn sự hướng dẫn tận tình của cô **Nguyễn Thị Minh Hỷ** và sự đóng góp ý kiến của các bạn,nhằm giúp cho đồ án được hoàn thành đúng yêu cầu. Mặc dù đã cố gắng tham khảo nhiều tài liệu nhưng trong quá trình thực hiện khó tránh khỏi sai sót và hạn chế. Rất mong nhận được ý kiến, đóng góp từ thầy cô và các bạn.

**Sinh viên thực hiện**

Nguyễn Thị Thư

Nguyễn Thị Anh Thư

**Lớp:16TCLC3**

MỤC LỤC

[LỜI NÓI ĐẦU 1](#_Toc8563051)

[I. LÝ THUYẾT VỀ PHÂN TÍCH CÚ PHÁP BẰNG PHƯƠNG PHÁP ĐỆ QUY KHÔNG QUAY LUI 3](#_Toc8563052)

[1. TÌM HIỂU CHUNG 3](#_Toc8563053)

[2. TRÌNH TỰ THỰC HIỆN 3](#_Toc8563054)

[a) Xây dựng văn phạm 3](#_Toc8563055)

[b) Xây dựng biểu đồ cú pháp 3](#_Toc8563056)

[c) Chương trình con biểu diễn cho biểu đồ cú pháp 3](#_Toc8563057)

[II. ÁP DỤNG PHÂN TÍCH CÚ PHÁP CHO NGÔN NGỮ C 7](#_Toc8563058)

[1) XÂY DỰNG VĂN PHẠM 7](#_Toc8563059)

[ Xây dựng văn phạm 7](#_Toc8563060)

[ Biến đổi văn phạm LL(1) 8](#_Toc8563061)

[2) XÂY DỰNG BIỂU ĐỒ CÚ PHÁP 9](#_Toc8563062)

[3) XÂY DỰNG CHƯƠNG TRÌNH CON PHỤ TRỢ 10](#_Toc8563063)

1. LÝ THUYẾT VỀ PHÂN TÍCH CÚ PHÁP BẰNG PHƯƠNG PHÁP ĐỆ QUY KHÔNG QUAY LUI
2. TÌM HIỂU CHUNG

* Là phương pháp phân tích cú pháp từ trên xuống
* Về mặt nguyên lý giống phương pháp tiên đóan
* Khác về lập trình: không tra bảng tiên đoán M mà vẫn mô phỏng trực tiếp
* Thay stack bằng sự đệ quy trong chương trình
* Một kí hiệu chưa kết thúc :được biểu diễn bằng một biểu đồ cú pháp
* Một biểu đồ cú pháp: biểu diễn bằng 1 chương trình con

1. TRÌNH TỰ THỰC HIỆN
2. Xây dựng văn phạm

* Xây dựng văn phạm chính quy
* Biến đổi từ văn phạm chính quy sang văn phạm LL(1) theo 3 quy tắc

1. Xây dựng biểu đồ cú pháp

* Kí hiệu chưa kết thưc đặt :
* Kí hiệu kết thúc đặt :
* Ví dụ : E -> TE’

E’

T

1. Chương trình con biểu diễn cho biểu đồ cú pháp
2. Sự kết hợp liên tiếp của các nút : sự kế tiếp của các đoạn chương trình tương ứng

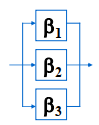
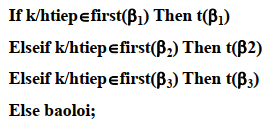
Ví dụ: E có đoạn chương trình t(E)

E’

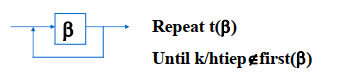
T

t(T); t(E’);

1. Sự rẽ nhánh tạo thành cấu trúc chọn

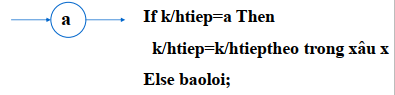
 

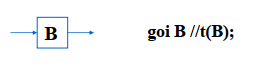
1. Lặp kiểm tra điều kiện sau



1. Lặp kiểm tra điều khiện trước



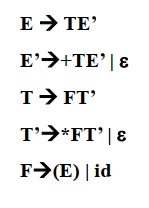




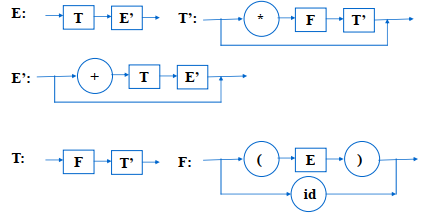
1. Thuật toán

* k/htiep: ký hiệu kết thúc
* function Dockh: trả về ký hiệu tiếp theo trong x
* Procedure Baoloi: cout lỗi, dừng
* Procedure B1: các chương trình con biểu diễn A∈Δ
* Procedure PTCP: đọc chuỗi x, nếu k/htiep=$ thì x đúng cú pháp else baoloi;

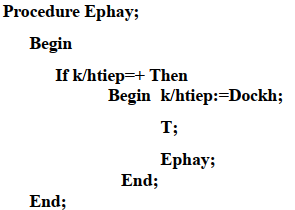
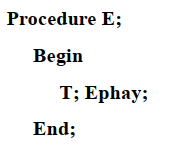
1. Ví dụ:

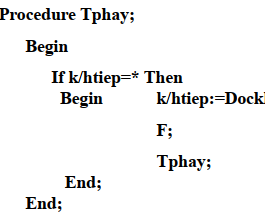
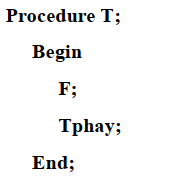


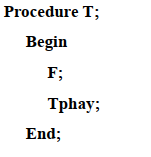
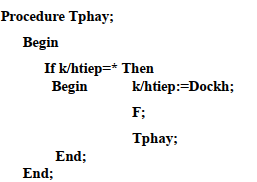
Biểu đồ cú pháp:

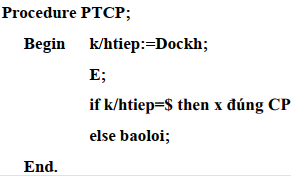
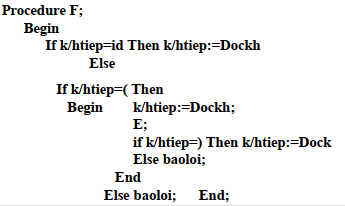


Chương trình con:







1. ÁP DỤNG PHÂN TÍCH CÚ PHÁP CHO NGÔN NGỮ C
2. XÂY DỰNG VĂN PHẠM

* Xây dựng văn phạm

G={ Σ, Δ,S,P } trong đó

Σ = token (từ phân tích từ vựng)

Δ={START,KB\_MAIN,KB’,HAM\_NMH,O,FTV,KH,KH’,TENTV,TEN,KIEU,SO,SN,STRING,STRING’,LK,B,MAIN,CT,BIEN,BIEN\_,BIEN\_’,IN,OUT,IF,LOOP,STEP,OP,BT,BT’,P\_SO,SN,S,D,SO’,OP’,TTTH,TTTG,TTG,TTL\_SIZE,TTLG,A,K,P\_TOAN,P\_TG,P\_G,I,SW,C,D,BR,BR’,MANG,GAN,LOIGOIHAM, GT}

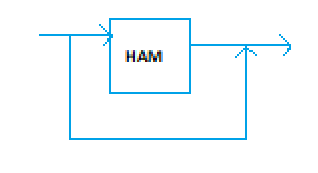
S={START}

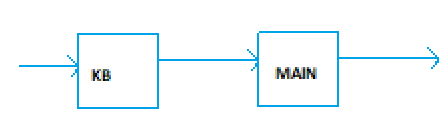
P={

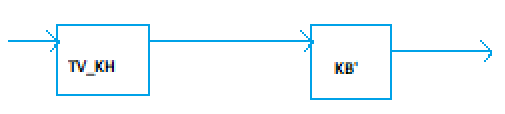
* START -> KB\_MAIN | KB\_MAIN HAM
* KB\_MAIN -> KB MAIN
* KB -> TV\_KH KB’
* KB’ -> KB | ε
* TV\_KH -> TV KH | TV
* TV -> #include<TENTV> |#define STRING SO| typedef STRING STRING
* TENTV -> STRING
* STRING -> a|..|z|A|..|Z|0|..|9|%|/|?...|STRING’
* STRING’ -> STRING| ε
* SO ->SN . S | SN
* SN -> D S
* D -> - |ε
* S -> 0|..|9|S’
* S’ -> S | ε
* KH -> BIEN | HAM \_ NMH | HANG |KH’
* KH’ -> KH |ε
* BIEN -> BIEN\_ ;
* BIEN\_ -> KIEU BIEN\_’
* KIEU -> int|short|float|double|long|long long |unsigned int|unsigned char |bool|byte|char
* BIEN\_’ -> TEN X
* X -> MANG | LK | O
* LK -> LK’ B
* LK’ -> TEN
* TEN->a TEN’| b TEN’| c TEN’|… |\_ TEN’
* TEN’->a TEN’| b TEN’| c TEN’|… |\_ TEN’| 0 TEN’|…| 9 TEN’| ε
* B -> , LK | ε
* MANG ->[ SO ] GAN
* GAN -> ={PT}
* PT -> STRING PT\_PP
* PT\_PP -> , PT | ε
* O -> = STRING
* HAM\_NHM -> KIEU STRING (STRING) F| void STRING(STRING) F
* F -> NHM | HAM
* NHM -> ;
* HAM-> { CT}
* HANG -> const TEN SO
* MAIN -> main ( ){ CT}
* CT ->BIEN| IN |OUT| LOOP|STEP|IF|OP|LOIGOIHAM|CT’
* CT’-> CT| ε
* IN -> scanf(“STRING”);
* OUT ->printf(“STRING”);
* LOOP -> while(BT){CT} | do {CT} while (BT); |for (BIEN; BT;OP) V
* V-> {CT} | CT
* BT -> STRING BT’ | (STRING BT’)
* BT -> P\_SO STRING | ε
* P\_SO -> < | > |=| != | >= | <= | ==
* STEP -> continue; | break;
* OP -> OP ‘ ;
* OP’ -> TEN TYPE |TTL\_SIZE
* TYPE -> TTTH | TTTG | TTG| TTLG
* TTTH -> = A
* A -> GT P\_TOAN | (GT P\_TOAN ) P\_TOAN
* GT -> TEN | SO
* P\_TOAN -> +A | -A| /A| % A| \*A| ε
* TTTG -> P-TG | P\_TG TEN
* P\_TG -> ++ | --
* TTG -> P\_G A
* P\_G -> += | -= | /= | \*= | %=
* TTL\_SIZE -> sizeof(STRING)
* TTLG -> BT K | (BT ) K
* K -> && TTLG | || TTLG | ! TTLG | ε
* IF -> I | SW
* I -> if (TTLG) I’
* I’ -> {CT} | CT I’’
* I’’ -> else I’ | ε
* SW -> switch (BT) { C D }
* C-> case (STRING) : CT BR
* BR -> BR’ C
* BR’-> break; | ε
* D -> default : CT |ε
* Biến đổi văn phạm LL(1)
* START -> KB\_MAIN | KB\_MAIN HAM
* START -> KB\_MAIN START’
* START’ -> HAM | ε
* TV\_KH -> TV KH | TV
* TV\_KH -> TV TV\_KH’
* TV\_KH’ -> KH | ε

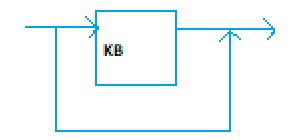
1. XÂY DỰNG BIỂU ĐỒ CÚ PHÁP

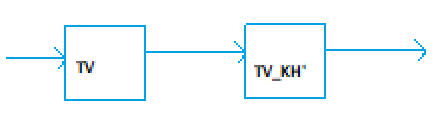
**START** : 

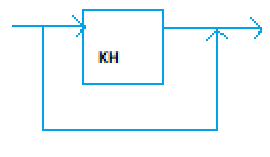
**START**’ : 

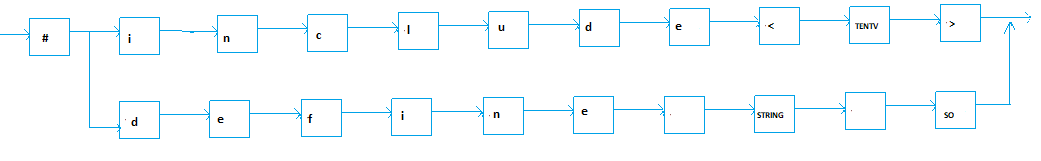
**KB\_MAIN:** 

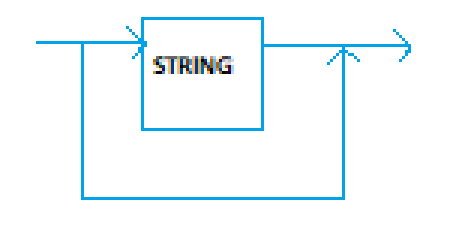
**KB:** 

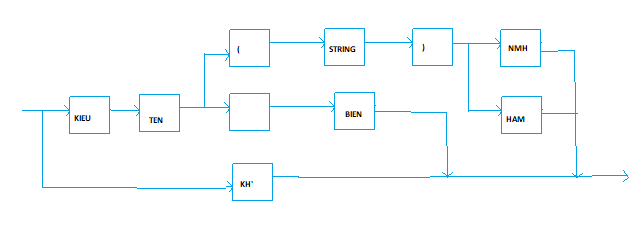
**KB’ :** 

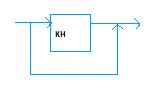
**TV\_KH :** 

**TV\_KH’ :** 

**TV:**

**TENTV:** 

**KH:** 

**KH’:**

BIEN\_

**BIEN:**

BIEN\_’

KIEU

**BIEN\_:**

**KIEU:**

…

**BIEN\_’:**

X

TEN

**X:**

MANG

LK

O

**LK:**

LK’

B

**LK’:**

TEN

**TEN:**

TEN’

…

**TEN:**

TEN’

…

**…**

LK

**B:**

ε

**MANG:**

GAN

SO

PT

**GAN:**

PT\_PP

STRING

**PT:**

PT

**PT\_PP:**

**O:**

STRING

STRING

STRING

KIEU

**HAM-NHM:**

F

NHM

**F:**

HAM

**NHM:**

CT

**HAM:**

SO

TEN

**HANG:**

**MAIN:**

CT

CT’

BIEN

**CT:**

IN

OUT

**…**

LOIGOIHAM

**CT’:**

CT

STRING

**IN:**

STRING

**OUT:**

**LOOP:**

CT

BT

CT

CT

OP

BIEN

BT

V

**V:**

CT

CT

STRING

BT’

**BT:**

BT’

STRING

STRING

P\_SO

**BT’:**

**P\_SO:**

**…**

**STEP:**

**OP’:**

BT’

STRING

TTL\_SIZE

TTTH

**TYPE:**

TTTG

TTG

TTLG

**TTTH:**

A

P\_TOAN

GT

**A:**

P\_TOAN

GT

TEN

**GT:**

SO

**P\_TOAN:**

A

…

TEN

P\_TG

**TTTG:**

P\_TG

**P\_TG:**

**TTG:**

P\_G

A

**P\_G:**

**…**

STRING

**TTL\_SIZE:**

**V:**

K

BT

BT

**P\_TOAN:**

TTLG

**IF:**

I

SW

I’

TTLG

**I:**

**I’:**

I’’

CT

CT

I’

**I’’:**

C

**SW:**

D

BT

**C:**

STRING

CT

BR

**BR:**

BR’

C

**BT’:**

CT

**BT’:**

1. XÂY DỰNG CHƯƠNG TRÌNH CON PHỤ TRỢ