**LÝ THUYẾT CHƯƠNG TRÌNH DỊCH**

**Câu 1: Chương trình dịch là gì? Các giai đoạn của chương trình dịch? Chương trình là gì?**

* **Chương trình dịch** là một chương trình thực hiện việc chuyển đổi 1 chương trình hay một đoạn chương trình con từ ngôn ngữ nguồn sang ngôn ngữ đích.
* **Ngôn ngữ nguồn:** là ngôn ngữ lập trình bậc cao ví dụ như C, C++, Java
* **Ngôn ngữ đích:** là ngôn ngữ lập trình bậc thấp Assembly, ngôn ngữ máy
* Chương trình dịch không có khả năng phát hiện được lỗi ngữ nghĩa và tự động biểu diễn đầu vào.
* Các giai đoạn trong chương trình dịch: gồm 2 pha là giai đoạn phân tích (ký trước) và giai đoạn tổng hợp (kỳ sau)
* Giai đoạn phân tích: gồm phân tích từ vựng, phân tích cú pháp, và phân tích ngữ nghĩa.
* Giai đoạn tổng hợp: gồm sinh mã trung gian, tối ưu mã trung gian và sinh mã đích.
* **Chương trình là** dãy các lệnh được tổ chức theo các quy tắc được xác định bởi ngôn ngữ lập trình cụ thể.

**Câu 2: Biên dịch và thông dịch**

* **Biên dịch** là chuyển từ chương trình nguồn sáng chương trình đích ở dạng mã máy như TurBo Pascal.
* **Thông dịch**: đọc chương trình từ nguồn theo lệch rồi phân tích rồi thực hiện nó. Chuyển từ ngôn ngữ nguồn sang ngôn ngữ dạng trung gian.
* Chương trình đọc ngôn ngữ trung gian thì được gọi là máy.

**Câu 3: Phân tích từ vựng, phân tích cú pháp, phân tích ngữ nghĩa?**

**Phân tích từ vựng:**

* Phân tích từ vựng: đọc chương trình nguồn sau đó lấy ra các từ vựng và xác định loại từ tố cho từ vựng đó.
* Từ tố chính là dạng cú pháp của từ vựng trong phân tích cú pháp ví dụ như tên, từ khoá, phép toán, …
* Mỗi loại từ tố có một luật mô tả, qua đó phân loại được từ vựng thuộc từ tố nào

Ví dụ if a > b then max: = a => có bao nhiêu loại từ từ tố và đó là từ tố nào?

**Phân tích cú pháp:**

* Tìm cấu trúc cú pháp của chương trình nguồn như biểu thức, phép gán, khai báo, câu lệnh, … để kiểm tra các cú pháp đã đúng hay chưa.
* Giai đoạn phân tích cú pháp dùng để nhóm các thẻ rừ của chương trình nguồn thành các đoạn văn phạm mà sau đó được trình biên dịch tập hợp ra thành phần.

**Phân tích ngữ nghĩa**

* Kiểm tra lỗi ngữ nghĩa: như kiểu dữ liệu, tên biến, sử dụng tên, …
* Thu nhận thông tin thuộc tính cho từ tố như số 🡪 giá trị, tên 🡪 hằng, biến , …
* Đồng thời kiểm tra kiểu trong phép gán, tính toán.

**Câu 4: Đầu vào, đầu ra của bộ phân tích từ vựng**

* Đầu vào của bộ phân tích từ vựng là mã nguồn cần phân tích, không có bất kì ràng buộc nào.
* Đầu ra của bộ phân tích từ vựng là tập các từ tố (token).

**Câu 5: Nêu định nghĩa văn phạm, ý nghĩa và các kí hiểu?**

* **Văn phạm** là một hệ thống gồm 4 thành phần **(**Σ, Δ, P, S)

**Trong đó:**

* Σ: là tập hợp các kí hiệu kết thúc được kí hiểu bởi các chữ cái viết thường như a, b, c, …
* Δ: là tập hợp các kí hiệu không kết thúc được kí hiệu bởi các chữ cái viết hoa như A, B, C.
* P: là tập hợp các luật sinh có dạng apha 🡪 beta
* S: là kí hiệu không kết thúc thuộc Δ và là kí hiệu bắt đầu.
* **Phân loại văn phạm:**
* Văn phạm phi ngữ cảnh (loại 2)
* Văn phạm chính quy (loại 3) dựa vào A 🡪 wB, A 🡪 w; A 🡪 Bw, A 🡪 w (ví dụ như: S 🡪 Aa; S 🡪 aA (được đoán nhận bởi automat hữu hạn, sinh ra ngôn ngữ chính quy)

**Câu 6: Văn phạm chính quy, ngôn ngữ chính quy, dẫn xuất của văn phạm đệ quy trái, đệ quy phải?**

* **Văn phạm chính quy là văn phạm mà có các luật sinh đều có dạng tuyến tính trái hoặc tuyến tính phải.**
* **Cho văn phạm G = (**Σ, Δ, P, S) thì dẫn xuất của văn phạm đệ quy trái hoặc đệ quy phải sẽ có dạng:
* A🡪aB; A🡪a (đệ quy phải do có kí hiệu không kết thúc B nằm ở phía bên phải)
* A🡪Ba; A🡪a (đệ quy trái do có kí hiệu không kết thúc B nằm ở phía bên trái)
* G được gọi là văn phạm chính quy và ngôn ngữ được sịnh ra bởi G là ngôn ngữ chính quy
* Văn phạm phi ngữ cảnh: là văn pham có dạng A 🡪 w | A Δ, w (ΣΔ)\*
* Văn pham tương đương: 2 văn phạm được gọi là tương đương nếu cùng sinh ra một ngôn ngữ L(G)

**Câu 7: Biểu thức chính quy?**

* Biểu thức chính quy là lựa chọn phổ biến để mô tả từ vựng.

Ví dụ: Cho ∑ = {a, b}

* Biểu thức chính quy a | b đặc tả {a, b}
* Biểu thức chính quy (a | b) (a | b) đặc tả tập hợp {aa, ab, ba, bb}. Tập hợp này có thể được đặc tả bởi biểu thức chính quy tương đương sau: aa | ab | ba | bb.
* Biểu thức chính quy a\* đặc tả {ε, a, aa, aaa, ...}
* Biểu thức chính quy (a | b)\* đặc tả {a, b, aa, bb, ...}. Tập này có thể đặc tả bởi (a\*b\*)\*.

Biểu thức chính quy a | a\* b đặc tả {a, b, ab, aab, ...} Hai biểu thức chính quy cùng đặc tả một tập hợp ta nói rằng chúng tương đương và viết r = s

**Câu 8: Trong biểu thức chính quy, toán tử \* có ý nghĩa gì?**

* Toán tử \* có ý nghĩa lặp lại 0, 1 hoặc nhiều lần ví dụ như a\* thì có nghĩa là lặp 0 đến nhiều lần kí tự a. Tương tự dấu + cũng thế nhưng là lặp 1 đến nhiều lần.
* Toán tử | có ý nghĩa là lựa chọn (hoặc).
* Toán tử (): có ý nghĩa là nhóm các thành phần
* Toán tử ? có ý nghĩa là lặp 0 hoặc một lần.

**Câu 9: Trong một trình biên dịch, gian đoạn phân tích từ vựng được thực hiện như nào?**

* Đọc chương trình nguồn từ trái qua phải để tách thành các thể từ tố (token).

**Câu 10: Một văn phạm tạo ra nhiều hơn một cây phân tích cú pháp cho cùng một chuỗi nhập thì được gọi là gì?**

* Được gọi là văn phạm nhập nhằng vì có thể nhiều cách để dẫn đến chương trình nguồn và nó có thể tạo ra nhiều hơn 1 cây phân tích cú pháp cho cùng 1 chuỗi nhập.
* Cách có thể xoá bỏ nhập nhằng bằng:
  + thêm vào 1 số kí hiệu không kết thúc
  + Chỉ cho phép sử dụng đệ quy trái hoặc đệ quy phải.

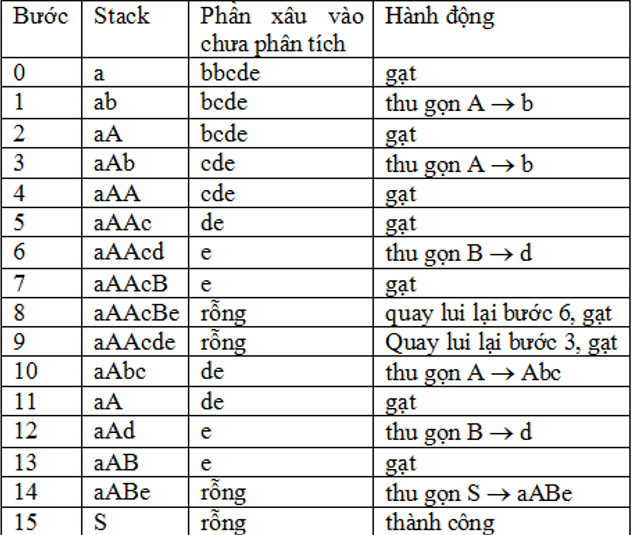
**Câu 11: Định nghĩa ô tô mát? các ô tô mát hữu hạn trạng thái FSA có vai trò gì trong chương trình dịch (otomat hữu hạn)?**

* **Định nghĩa**: otomat (cách gọi khác: máy tự động) là thiết bị có thể thực hiện công việc mà không cần sự can thiệp của con người.
* Tại sao lại được như vậy: tại vì nó hoạt động dựa trên một số quy tắc và trên các quy tắc do người lập trình tạo ra để nó hoạt động theo ý muốn của mình.
* **Thành phần của otomat:**
* Thiết bị đầu vào (input file): nơi để nhập chuỗi, nó được chia thành các ô, mỗi ô sẽ chứa 1 kí hiệu.
* Input mechanism (cơ cấu nhập): là bộ phận đọc chuỗi nhập input file từ trái sang phải
* Bộ nhớ tạm: là thiết bị bao gồm một số không giới hạn các ô nhớ, mỗi ô chứa một kí hiệu từ một bảng chữ cái, otomat có thể đọc và thay đổi nội dung của các ô nhớ.
* Đơn vị điều khiển: giúp chuyển trạng thái
* **Vai trò của otomat hữu hạn trạng thái FSA trong chương trình dịch:** có khả năng đoán nhận ngôn ngữ dễ dàng, được ứng dụng rất lớp trong việc xây dựng ngôn ngữ lập trình, dùng acceppter để định nghĩa một vài thành phần của ngôn ngữ lập.

**Câu 12: Mô tả thuật toán phân tích cú pháp bottom-up. Thuật toán thành công khi nào? Được dựa trên ý tưởng nào?**

* **Ý tưởng: x**ây dựng cây phân tích cú pháp theo lỗi nút lá đến gốc, sử dụng stack S để chứa kí hiệu của văn phạm cần phân tích.
* **Mô tả:**
* Bước 1: gạt ký hiệu đầu tiên của xâu x vào Stack
* Bước 2: lặp xét mọi xâu w có thể trên đỉnh stack ví dụ stack có abcd => mọi xâu có thể là d, cd, bcd, abcd
* Nếu tồn tại A 🡪 w thì thực hiện thu gọn: thu gọn bằng cách lấy tất cả ký hiệu của xâu w trong stack sau đó đẩy kí hiệu A vào stack. Trong trường hợp có nhiều xâu w cùng thoả mãn thì đánh số lần lượt
* Nếu không thể thu gọn thì gạt ký tự tiếp theo trên xâu x vào Stack sau đó lại thực hiện bước 2
* Nếu mà gạt hết các kí tự trên x mà trong stack không phải còn S thì quay lui địa chỉ sau cùng mà ở đó đã tiến hành thu gọn
* **Thuật toán thành công** khi mà đã gạt hết các ký hiệu trên xâu vào và trong stack chỉ còn 1 ký hiệu S. khi đó ta kết luận xâu được đoán nhận, ngược lại thì xâu không được đoán nhận.
* **Ví dụ cho văn phạm S → aABe; A** → **Abc | b; B** → **d**

Xâu vào: abbcde



**Câu 13: Phương pháp Shift reduce là phương pháp gì?**

* Phương pháp shift reduce là phương pháp xây dựng một cây phân tích cú pháp cho một chuỗi nhập bắt đầu từ nút lá về nút gốc.
* Thu gọn chuỗi w thành 1 kí hiệu bắt đầu cho văn phạm

Ví dụ: Cho văn pham: S 🡪 aABe; A 🡪 Abc | b; B🡪 d

Chuỗi nhập: abbcde

abbcde

aAbcde

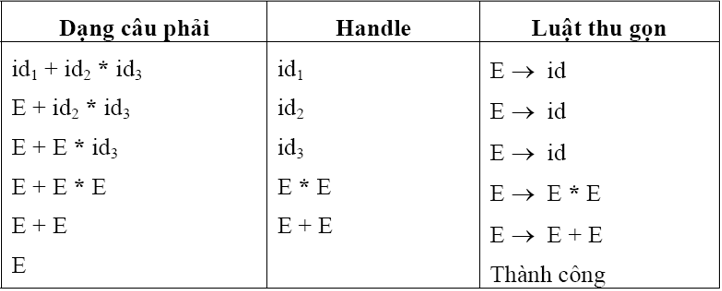
aAde

aABe

S

Kiểm tra lại: S => aABe => aAde => aAbcde => abbcde (chính xác)

**Ví dụ 2: Mô tả phân tích cú pháp cho câu nhập id1 + id2 \* id3**



**Câu 14: định nghĩa frist, trình bày cách tính frist**

* **Định nghĩa:** Giả sử là 1 chuỗi các kí hiệu văn phạm, First() là tập hợp các ký hiệu kết thúc mà nó bắt đầu 1 chuỗi dẫn xuất trừ
* **Cách tính:**

1. **Nếu X là kí hiệu kết thúc (là các chữ cái thường, toán tử hoặc epsilon) thì frist(X) = X**

Ví dụ: A 🡪 x Thì frist(A) = x; A🡪 epsilon thì frist(A) = epsilon; A 🡪 + thì frist(A) = +

1. **Nếu tồn tại luật sinh X** 🡪 **YY1Y2**

* Thì đơn giản là ta tính frist(Y) trước nếu frist(Y) không chứa epsilon thì frist (YY1Y2) = frist(Y) khi đó sẽ thêm tất các ký hiệu kết thúc # epsilon vào frist (X)

Ví du: A 🡪 aBD thì tính Y chính là a thì frist(a) = {a} khi đó thêm a vào frist(A)

* Nếu mà frist(Y) chứa epsilon thì tính đến frist(Y1) rồi thêm các kí hiệu kết thúc khác epsilon của Frist(Y1) vào frist(X)
* Nếu mà fist(Y) giao với frist(Y1) là epsilon thì thêm tất cả kí hiệu kết thúc khác epsilon của frist(Y2). Sau đó thêm epsilon nếu các frist giao nhau bởi epsilon

**Câu 15: Định nghĩa, Cách tính follow**

**Định nghĩa:** Follow(A): Với A là ký hiệu của kết thúc. Là tập hợp các ký hiệu kết thúc a mà nó xuất hiện ngay sau A (bên phải của A) trong 1 dạng câu nào đó. Tức là 1 tập ký hiệu kết thúc a, sao cho tồn tại 1 dẫn xuất dạng S=>\* Aa

**Cách tính:**

* Nếu S là kí hiệu bắt đầu thì thêm S vào follow(S)
* Nếu có một luật sinh A→ αBβ thì thêm mọi phần tử khác ε của FIRST(β)vào trong FOLLOW(B).
* Nếu có luật sinh A→ αB hoặc A→ αBβ mà ε ∈ FIRST(β) thì thêm tất cả các phần tử trong FOLLOW(A) vào FOLLOW(B)

**Câu 16: Giai đoạn cuối của trình biên dịch có tên gọi là gì? Nó thực hiện công việc gì?**

* Giai đoạn cuối của trình biên dịch có tên gọi là sinh mã
* Thực hiện tạo ra mã máy hoặc các mã assembly
* Nó thực hiện các hoạt động quản lý bộ nhớ, gán đăng ký và tối ưu hoá cụ thể cho máy

**Câu 17: thế nào là văn phạm tăng cường**

G là văn phạm với ký hiệu bắt đầu S, ta thêm 1 ký hiệu bắt đầu mới S’, và luật sinh S’🡪S để được văn phạm mới G’ => văn phạm tăng cường