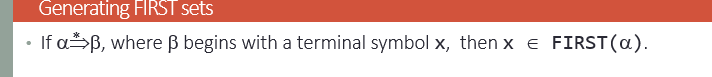
如果a能一直推到B，直到B的开头是terminal symbol



那么这个terminal x属于 a的first

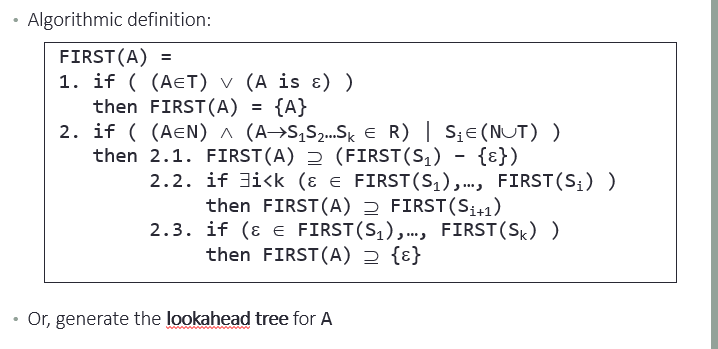
同理这一路上所有non terminal 的开头都应该是这个x

例如

A->BC ，那么A的开头肯定和B相同

B->EF 那么B的开头肯定和E相同

E->das 那么ABE的开头terminal都是d



算法描述，如果A是terminal或者A是空集，那么FIRST (A)={A} //terminal的first是自己

如果

如果A是Non-terminal，且A->S1S2…. //Si可以是NUT

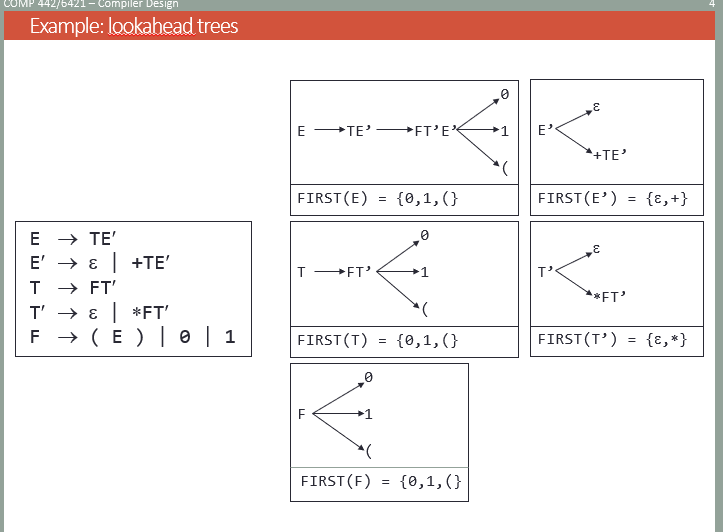
三种情况，

1：S1的first减去e必然是 FIRST (A)的一部分

2：如果存在一个i，s1到Si的first选择都是e //例如 S1->S2SASDF | e ，那么就可以省略S1，那么FIRST(Si+1)属于FIRST (A)的一部分

3：如果S1 一直到SK全部存在e，那么e属于FIRST(A)

或者直接生成一个lookahead tree



就是不停地推导第一个NON-TERMINAL 元素

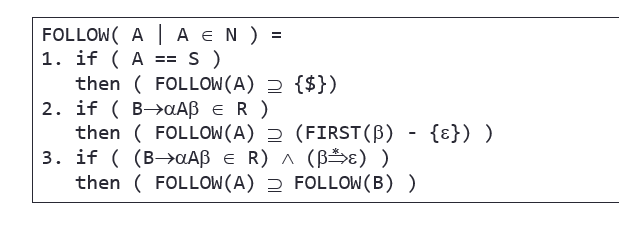
E->TE' 然后展开T ,TE'->FT'E' 然后展开F->0.1.(

s

生成follow set

首先我们要知道follow set 的定义

紧跟在后的第一个terminal 的集合， 且不包括e空串



算法描述

1.如果A==S， 即A是开头，那么在FOLLOW A中添入DOLLAR 符，

2.如果是B->aAb的形式

那么first(b)-空串属于 FOLLOW(A) //好理解，这里与B或α无关，而是后面两个的关系，β是紧跟A的

3.如果是B->αAβ的形式且β最终能成为空串， 或者 就是B->αA的形式

那么followB是followA的一部分，为什么？

因为假设词语是BC,然后B能推出αA，那么就是αAC，那么必然两个人的follow都是first of C

注意first可以有空串e

FOLLOW不能有空串

Recursive descent predictive parsing

Method：

首先要建立FIRST 与FOLLOW set

然后对于每一个non-terminal 我们要有一个对应function //编写代码方面

然后对于每个function:选择建立在右手边的first set //绝大多数情况 //想想构建树

而对于类似A->e的可以选择的路径，我们就可以在Select可选范围中加入A的follow

例如 S->AE

A->BD FIRST(a,b)

A-> CE FIRST(c,d)

A->e空串

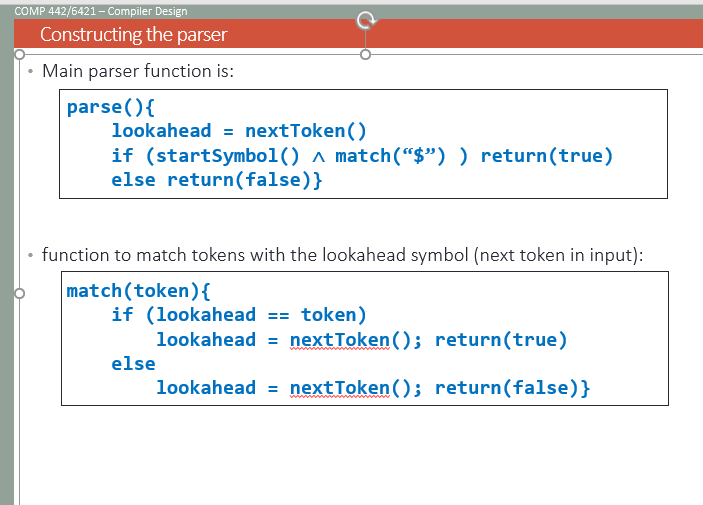
E->r

假如Input token为a或者c，要在两条first路径中选择，

而如果Input token是r，就要让A->e，选择空串，因为他的follow有r

如果一条valid path都没有，return false

不然return true



主要的

先找第一个token

然后激活startSymbol，

这个startSymbol实际上才是我们的具体程序，

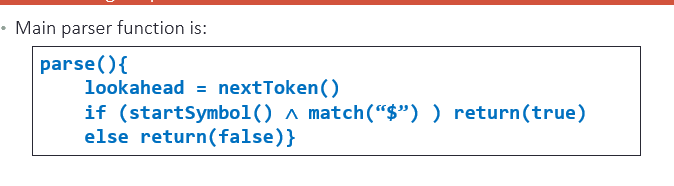
例如有一个S->ABCDEFG

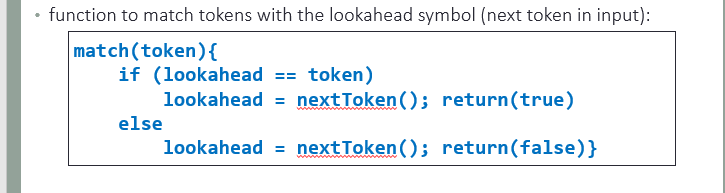
他会遍历所有的ABCDEFG直到底部，如果中间匹配不了，就return error

如果直到底部成功了，那么就看末尾是不是dollar符。

如果两个判断条件都对

说明，我们不但让S->ABCDEFG全部匹配到具体的terminal，而且还成功匹配到底部也就是生成了一个句子

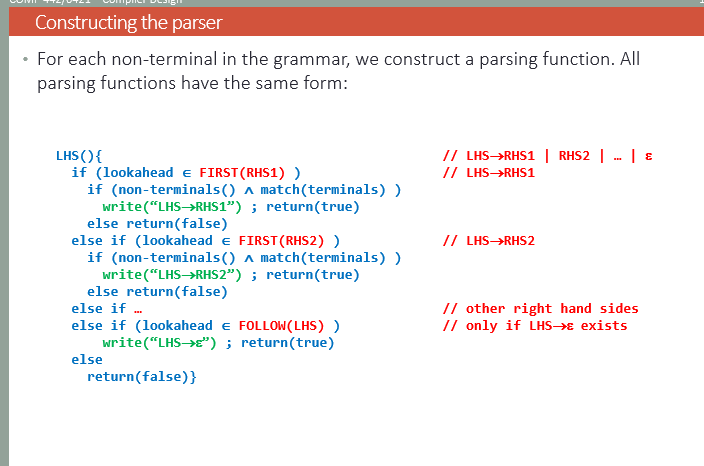




match token //针对terminal

如果当前input 为token，那么说明Match， 输入下一个token , return true

不然就输入下一个token return false

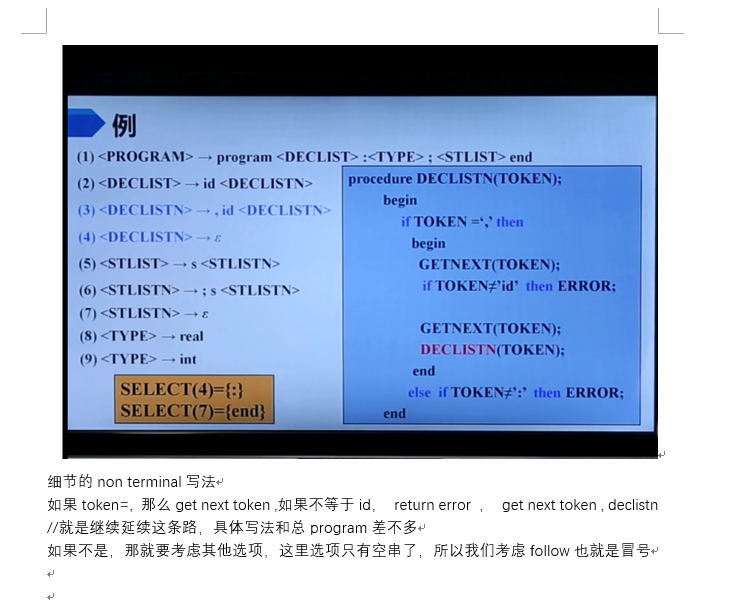


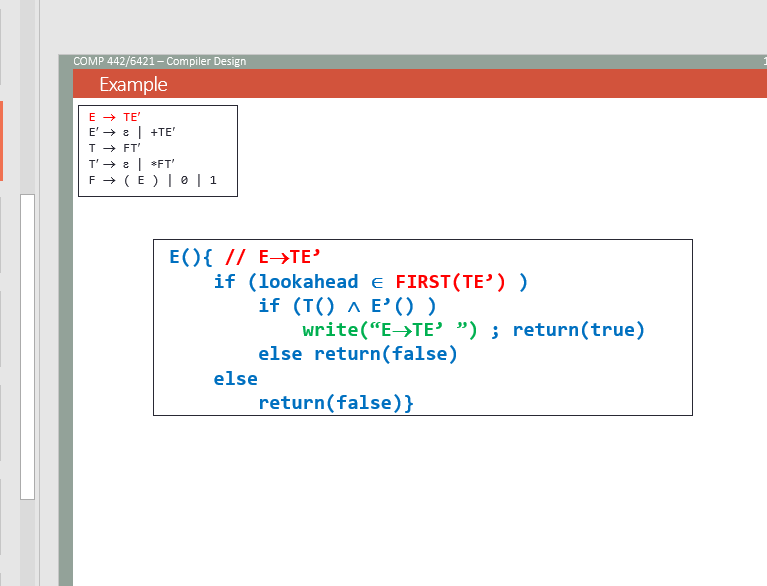
对于每一个non-terminal，我们都要构建一个分析function

首先把first的都排查一遍

如果input (INPUT肯定是terminal) 且在RHS1的first里选RHS1，..如果在RHS2里选RHS2

如果都没有，属于左手的follow lhs，那么就选择LHS->e





具体的

第一步

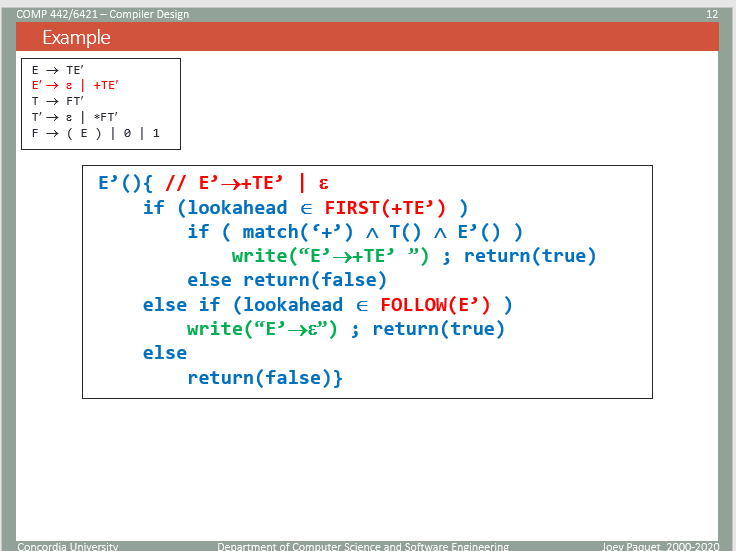
如果LOOKAHEAD属于TE'的first

激活T() E'() RERURSION, 他们如果顺利写完，写下E->TE'

如果T()E()没顺利写完，return false

不然直接return false，代表input不属于这条路径

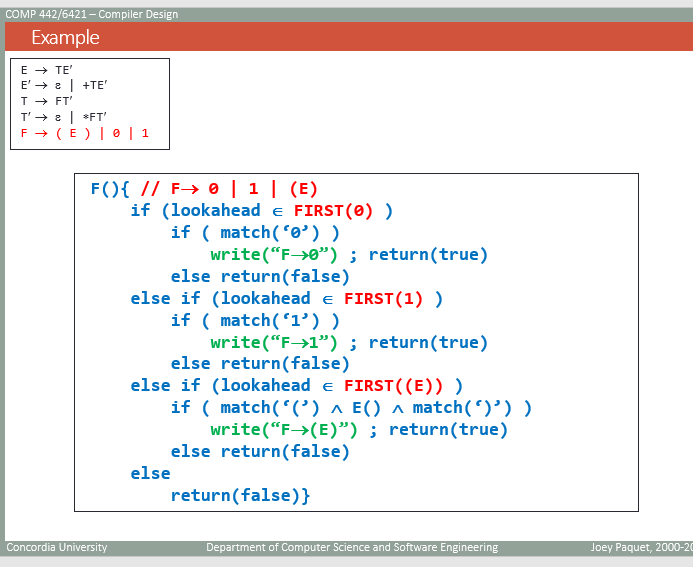
这条规则就写好了



第一个if循环大同小异

terminal需要match 加号

然后如果属于follow，让他等于空集，就算循环完成了·，return true



first 0

first 1都是Match，没啥好说的

总结，不是空的就看是不是在 右侧first集里，是空的就看看match follow左侧

如果是terminal， 使用match function，如果是Non terminal，在if里面recursive call