

程序定义：

$\langle program \rangle \rightarrow \langle segment \rangle \langle program \rangle \mid \epsilon$

program表示主程序，segment表示语句片段

segment主要包括变量声明定义和函数声明定义等

声明用extern（主要用于外部声明）

程序段定义：

$\langle segment \rangle \rightarrow \text{extern } \langle type \rangle \langle def \rangle \mid \langle type \rangle \langle def \rangle$

$\langle type \rangle \rightarrow \text{int} \mid \text{char} \mid \text{void}$

type表示基本类型

def表示定义

变量定义：

$\langle vardef \rangle \rightarrow \text{id} \langle norvardef \rangle \mid * \text{id} \langle init \rangle$

$\langle norvardef \rangle \rightarrow [\text{num}] \mid \langle init \rangle$

$\langle init \rangle \rightarrow = \langle expr \rangle \mid \epsilon$

$\langle varlist \rangle \rightarrow, \langle vardef \rangle \langle varlist \rangle \mid ;$

vardef表示变量定义，区分了普通变量与指针变量定义

norvardef表示数组定义or变量初始化，这里不支持数组初始化

init表示初始化，expr表示表达式，具体放在后文来说

varlist表示支持同时定义多个变量，可以用表示变量定义

函数定义：

函数定义：

$\langle fundef \rangle \rightarrow \text{id}(\langle para \rangle) \langle funtail \rangle$

$\langle funtail \rangle \rightarrow ; \mid \langle funbody \rangle$

fundef表示函数：函数名(参数列表) 函数声明/定义

funtail是表示函数为声明还是定义

参数定义：

$\langle para \rangle \rightarrow \langle type \rangle \langle paradef \rangle \langle paralist \rangle \mid \epsilon$

$\langle paradef \rangle \rightarrow * \text{id} \mid \text{id} \langle paratail \rangle$

$\langle paratail \rangle \rightarrow [\text{num}] \mid \epsilon$

$\langle paralist \rangle \rightarrow , \langle type \rangle \langle parade f \rangle \langle paralist \rangle | \epsilon$

para表示参数的总定义

parade f表示参数是指针or变量(数组)

paratail表示是否为数组

parallist表示定义多个参数

函数体定义:

$\langle funbody \rangle \rightarrow \{ \langle funprogram \rangle \}$

$\langle funprogram \rangle \rightarrow \langle localdef \rangle \langle funprogram \rangle | \langle statement \rangle \langle funprogram \rangle | \epsilon$

$\langle localdef \rangle \rightarrow \langle type \rangle \langle vardef \rangle \langle varlist \rangle$

funbody表示函数体定义

funprogram表示函数的程序，可以有局部变量定义，也可以有语句

localdef就是变量定义，可以用上文的表示

statement放在后文来说

变量总定义:

$\langle def \rangle \rightarrow id \langle deftail \rangle | *id \langle init \rangle \langle varlist \rangle$

$\langle deftail \rangle \rightarrow \langle norvardef \rangle \langle varlist \rangle | (\langle para \rangle) \langle funtail \rangle$

为什么不能直接 $\rightarrow |$ 呢?

因为这里涉及左公因子id，因此需要展开后进行拆分

表达式定义:

运算符	含 义	优先级	结合性
=	赋值	10	右结合
	逻辑或	9	左结合
&&	逻辑与	8	左结合
> < >= <= == !=	大于、小于、大于等于、小于等于、等于、不等于	7	左结合
+ -	加法、减法	6	左结合
* / %	乘法、除法、取模	5	左结合
! - & * ++ --	逻辑非、取负、取址、指针、前置++、前置--	4	右结合
++ --	后置++、后置--	3	右结合
()	括号	2	左结合
[] ()	数组索引、函数调用	1	左结合

构造表达式语法时，需要从运算符优先级从低到到来进行考虑。(上表中，优先级数字越低表示优先级越高)

$\langle assexpr \rangle \rightarrow \langle oorexpr \rangle \langle asstail \rangle$

$\langle asstail \rangle \rightarrow = \langle orrexpr \rangle \langle asstail \rangle | \epsilon$

=

$\langle orrexpr \rangle \rightarrow \langle aandexpr \rangle \langle oortail \rangle$

$\langle oortail \rangle \rightarrow \mid \mid \langle aandexpr \rangle \langle oortail \rangle \mid \epsilon$

$\mid \mid$

$\langle aandexpr \rangle \rightarrow \langle cmpexpr \rangle \langle aandtail \rangle$

$\langle aandtail \rangle \rightarrow \&\& \langle cmpexpr \rangle \langle aandtail \rangle \mid \epsilon$

$\&\&$

$\langle orexpr \rangle \rightarrow \langle xorexpr \rangle \langle ortail \rangle$

$\langle ortail \rangle \rightarrow or \langle xorexpr \rangle \langle ortail \rangle \mid \epsilon$

\mid

$\langle xorexpr \rangle \rightarrow \langle andexpr \rangle \langle xortail \rangle$

$\langle xortail \rangle \rightarrow xor \langle xorexpr \rangle \langle xortail \rangle \mid \epsilon$

\wedge

$\langle andexpr \rangle \rightarrow \langle cmpexpr \rangle \langle andtail \rangle$

$\langle andtail \rangle \rightarrow and \langle cmpexpr \rangle \langle andtail \rangle \mid \epsilon$

$\&$

$\langle cmpexpr \rangle \rightarrow \langle aloexpr \rangle \langle cmptail \rangle$

$\langle cmptail \rangle \rightarrow \langle cmps \rangle \langle aloexpr \rangle \langle cmptail \rangle \mid \epsilon$

$\langle cmps \rangle \rightarrow > \mid > = \mid < \mid < = \mid == \mid !=$

比较运算符

$\langle aloexpr \rangle \rightarrow \langle item \rangle \langle alotail \rangle$

$\langle alotail \rangle \rightarrow \langle alos \rangle \langle item \rangle \langle alotail \rangle \mid \epsilon$

$\langle alos \rangle \rightarrow + \mid -$

+ - 运算符

$\langle item \rangle \rightarrow \langle factor \rangle \langle itemtail \rangle$

$\langle item \rangle \rightarrow \langle its \rangle \langle factor \rangle \langle items \rangle \mid \epsilon$

$\langle its \rangle \rightarrow * \mid / \mid \%$

* / %运算符

$\langle factor \rangle \rightarrow \langle lop \rangle \langle factor \rangle \mid \langle val \rangle$

$\langle lop \rangle \rightarrow ! \mid - \mid \& \mid * \mid + \mid - -$

逻辑非 取负 取址 指针 左自增 左自减

$\langle val \rangle \rightarrow \langle element \rangle \langle rop \rangle$

$\langle rop \rangle \rightarrow + + \mid - -$

右自增 右自减

$\langle element \rangle \rightarrow id \langle idexpr \rangle \mid (expr) \mid literal$

$\langle idexpr \rangle \rightarrow [\langle expr \rangle] \mid (\langle realarg \rangle)$

$\langle realarg \rangle \rightarrow \langle expr \rangle \langle arglist \rangle \mid \epsilon$

$\langle arglist \rangle \rightarrow , \langle expr \rangle \langle arglist \rangle \mid \epsilon$

元素可以是变量，数组，函数，括号表达式以及常量

$\langle literal \rangle \rightarrow num \mid char \mid str$

常量包括数字，字符和字符串

$\langle expr \rangle \rightarrow \langle assexpr \rangle$

$\langle altxpr \rangle \rightarrow \langle expr \rangle \mid \epsilon$

表达式从assexpr开始

altexpr可以让表达式为空，常用语for语句中

语句定义：

语句要分为表达式语句，循环语句，判断语句，break，continue，return以及自定义语句等等。

$\langle statement \rangle \rightarrow \langle altxpr \rangle ; \mid \langle whilestat \rangle \mid \langle forstat \rangle \mid \langle dowhilestat \rangle$

$\langle statement \rangle \rightarrow \langle ifstat \rangle \mid \langle switchstat \rangle \mid \langle secloudstat \rangle \mid break ; \mid continue ; \mid return ;$

$\langle statement \rangle \rightarrow \langle readstat \rangle \mid \langle writestat \rangle$

循环语句定义：

$\langle whilestat \rangle \rightarrow while(\langle altxpr \rangle) \langle funbody \rangle$

$\langle dowhilestat \rangle \rightarrow do \langle funbody \rangle while(\langle altxpr \rangle) ;$

while和do while形式还是比较清晰的

$\langle forstat \rangle \rightarrow for(\langle forinit \rangle ; \langle altxpr \rangle ; \langle altxpr \rangle) \langle funbody \rangle$

$\langle forinit \rangle \rightarrow \langle localdef \rangle \mid \langle altxpr \rangle$

这里就可以体现出altexpr->expr|e 的简便性了

判断语句定义：

$\langle ifstat \rangle \rightarrow if(\langle altxpr \rangle) \langle funbody \rangle \langle elsestat \rangle$

$\langle elsestat \rangle \rightarrow else \langle funbody \rangle \mid \epsilon$

if语句的定义

$\langle switchstat \rangle \rightarrow switch(\langle expr \rangle) \{ \langle casestat \rangle \}$

$\langle casestat \rangle \rightarrow case \langle literal \rangle : \langle funprogram \rangle \langle casestat \rangle \mid default : \langle funprogram \rangle$

switch语句的定义

其他语句定义:

$\langle \textit{secloudstat} \rangle \rightarrow \textit{secloud}(\textit{num})$

`secloud(x)`是一个自定义函数, 随机返回123 57 152 110这四个数

haha just for fun!

$\langle \textit{readstat} \rangle \rightarrow \textit{read}(\textit{num})$

$\langle \textit{writestat} \rangle \rightarrow \textit{write}(\textit{id})$

目前只支持读入数字和输出数字, 具体实现以后扩展

即id的类型要求为int型