**Context Free Grammar**

* Prog start body exit
* Start dt main() {
* body stmt
* exit }
* Stmt stmt\_list stmt | E
* Stmt\_list dec\_stmt | cond\_stmt | print\_stmt | dec\_assign\_stmt | input\_stmt | loop\_stmt | assign\_stmt | func\_stmt | return\_stmt |null\_stmt | call\_stmt | array\_stmt | comnt\_stmt
* dec\_stmt dt var punc
* dec\_assign \_stmt dt assign\_stmt
* assign\_stmt var = numb punc
* func\_stmt dt ( d\_stmt ) { stmt }
* cond\_stmt if\_stmt |switch\_stmt
* return\_stmt return var punc
* comnt\_stmt // string **|** /\* string \*/
* null\_stmt break punc | exit punc
* array\_stmt dt arr[num] = {input}
* loop\_stmt for\_loop | while\_loop | dowhile\_loop
* print\_stmt printf (“string”) punc | printf (“string” , var) punc | cout>> “string” punc
* Input\_stmt scanf (“string” , var ) punc | cin>> var puc
* call\_stmt var = var ( var ) punc
* for\_loop for (assign\_stmt con punc inc\_dec \_stmt)
* Inc\_dec\_stmt inc\_stmt | dec\_stmt
* Inc\_stmt var oprt | oprt var | var + digit
* dec\_stmt var oprt | oprt var | var - digit
* while\_stmt while ( con ) { stmt }
* dowhile\_stmt do { stmt } while (con) punc
* d\_stmt dt var , d\_stmt | dt var d\_stmt | E
* If\_stmt if (con) stmt else stmt **|** if (con) { stmt } else { stmt }
* con alpha relop alpha | alpha relop alpha log\_stmt alpha relop alpha
* Input “var” , input | digit , input
* log\_stmt && **|** ||
* relop > **|** < **|** == **|** <= **|** > = **|** !=
* String literal\_string
* dt int | float | char | void
* var id |id , var |&id | &id , var |E
* Punc : | ;
* Id letteralpha
* alpha letter | digit | alpha
* letter a**|**A**|**b**|**B| . . . |z|Z
* numb digits frac exp
* digits digitdigits | E
* frac **.** digits | E
* Exp E (+ | - ) digits | E
* digit 0**|**1**|** . . . **|**9
* oprt + **|** - **|** \* **|** / **|** ++ **|** - -