No. 12

Date. 先给 S.List 定义时长度设为 10000，L.List 也是.

S'→S　for(int i=0; i<S.lena; i++)

　　cout<< S.List[i] << ' ';

S→(L)　S.len = L.len +2;

　　S.lena = L.lena;

　　for (int i=0; i<S.lena; i++)

　　S.List[i] = L.List[i]+1;

S→a　S.len =1;

　　S.lena =1;

　　S.List[0] =1;

L→L1,S　~~L.len = L1.len + S.len +1~~;

　　for(int i= L1.lena; i< L1.lena + S.lena; i++)

　　L.List[i] = L1.len +1 + S.List[i- L1.lena];

　　L.len = L1.len +1 + S.len;

　　L.lena = L1.lena + S.lena;

L→S　L.len = S.len;

　　L.lena = S.len;

　　for(int i=0; i<S.lena; i++)

　　L.List[i] = ~~S.List~~ S.List[i];

len为串长，lena为a个数，List中存a以原序排序后a的位置

No.

Date.

4.14

$S' \to P$ cout << P.val;

(1) $P \to D$ P.val = D.val;

$D \to D_1; D_2$ D_1.val = D_1.val + D_2.val;

$D \to id : T$ D.val = 1;

$D \to proc\ id; D_1; S$ D.val = 1 + D_1.val;

val 指 id 个数

for(int i=0; i< P.id_len; i++)

(2) $S' \to P$ cout << P.List[i];

$P \to D$ for(int i=0; i< D.id_len; i++)

P.List[i] = D.List[i];

P.id_len = D.id_len;

$D \to D_1; D_2$ for(int i=0; i< D_1.id_len; i++)

D.List[i] = D_1.List[i] + 1;

for(int i = D_1.id_len; i < D_1.id_len + D_2.id_len; i++)

D.List[i] = D_2.List[i - D_1.id_len]++;

~~$D \to id : T$~~ D.id_len = D_1.id_len + D_2.id_len;

$D \to id : T$ D.id_len = 1;

D.List[0] = 0;

$D \to proc\ id; D_1; S$ D.List[0] = 0;

for(int i=0; i< D_1.id_len; i++)

D.List[i+1] = D_1.List[i];

D.id_len = D_1.id_len + 1;

id_len 为有几个 id

List 为 id 顺序排列下的嵌套深度