Lengkapilah parser yang telah anda buat pada praktikum 1 hingga 6 dengan symbol table sehingga pada proses parsing dapat dilakukan pemeriksaan suatu identifier yang digunakan dalam program telah dideklarasikan sebelumnya atau tidak!

Untuk symbol table gunakan linked-list dan metoda pengaksesannya LIFO !

```
#define VGLOBAL 1   /* global variable  */
#define VLOCAL  2   /* local variable   */
#define PNAME   3   /* procedure name   */
#define FNAME   4   /* function  name   */

typedef struct _stack {
  char  key[50];        /*  string of token */
  char  type;           /*  VGLOBAL | VLOCAL | PNAME | FNAME */
  int   address;        /*  reserved for code generator */
  int   nparam;         /*  parameter number(s) if type := PNAME | FNAME */
  struct _stack *next;
}  stack_t;


stack_t * push(stack_t *list,char *name,char type,int addr);
stack_t * pop(stack_t *list);
stack_t * search(stack_t *list,char *name);
```

Perlu diperhatikan bahwa setiap deklarasi identifier, baik variable maupun procedure, data identifier tersebut harus di-push kedalam stack. Namun demikian, khusus untuk variable lokal, setiap keluar dari pemrosesan inblok, semua variable lokal tersebut harus di-pop dari stack.

Testing Code :

Input Source

```
program example71;
var i,j,k;
begin
   i := 20;
   j := 40;
   k := (30 + i) * (80 –j div 4)
end.
```

```
program example72;
var n,x;
procedure prime;
var m;
begin
   m:=x div 2;
   while x <> (x div m) *m do m:=m-1;
   if m=1 then write(x)
end;
```

```
begin
  read(n);
  while i<n do begin
    x:=n;
    prime;
    n:=n-1;
  end
end.
```

```
program example73;
var n,temp;
procedure fact(n);
begin
   if n <= 1 then temp:=1
   else begin
          fact(n-1);
          temp := temp * n
        end
end;
begin
  read(n);
  fact(n);
  write(temp)
end.
```