```
node * parse_expression(lexan & LEX)
{ LEX.nextlex();
  if (LEX.kind == LX_number)
    return new node(N_integer, LEX.intvalue);
  else if (LEX.kind == LX_variable)
    return new node(N_variable, LEX.syminfo);
  else
    LEX.error("expecting expression, found " + LEX.form); }


node * parse_statement(lexan & LEX)
{ LEX.nextlex();

  if (LEX.kind == LX_RW_print)
  { node * r = new node(N_printstmt);
    r->subtree.push_back(parse_expression(LEX));
    return r; }

  else if (LEX.kind == LX_RW_when)
  { node * r = new node(N_whenstmt);
    r->subtree.push_back(parse_expression(LEX));
    r->subtree.push_back(parse_statement(LEX));
    return r; }

  else
    LEX.error("expecting statement, found " + LEX.form); }


int main()
{ iosystem IO(cin);
  symboltable ST;
  lexan LEX(IO, ST);
  ST.enter("var", LX_RW_var);
  ST.enter("when", LX_RW_when);
  ST.enter("print", LX_RW_print);
  node * r = parse_statement(LEX);
  print(r); }


$ lang2
print cat
printstmt
   variable syminfo=0x804e080

$ lang2
when x print y
whenstmt
   variable syminfo=0x804e080
   printstmt
      variable syminfo=0x804e0b0

$ lang2
when x when y print z
whenstmt
   variable syminfo=0x804e080
   whenstmt
      variable syminfo=0x804e0b0
      printstmt
         variable syminfo=0x804e0e0
```