```
program primePrinter
   var k : int;
   var count : int;
   var num : int;
   var isPrime : bool;
   var i : int;


   read(k);

   count := 0;
   num := 2;

   WHILE (count < k) begin
      isPrime := true;

      i := 2;

      WHILE (i * i <= num)
      begin
         IF (num % i = 0)
         begin
            isPrime := false;
         end;
         i := i + 1;
      end;

      IF (isPrime = true)
      begin
         write(num);
         count := count + 1;
      end;

      num := num + 1;
   end;
end
```

program ::="program" identifier declist stmtlist "end"
program primePrinter
declist ::= declaration | declaration ";" declist
declaration ::= "var" identifier ":" type
type ::="int" |"bool"

stmtlist ::= stmt | stmt ";" stmtlist
stmt ::= assignment | iostmt | ifstmt | whilestmt
iostmt ::= "read" "(" identifier ")" ";" | "write" "(" expression ")" ";"
stmt ::= assignment | iostmt | ifstmt | whilestmt
assignment ::= identifier ":=" expression ";"
whilestmt ::= "WHILE" "(" condition ")" "begin" stmtlist "end"
condition ::= expression relational_op expression
expression ::=term | expresion "+" term | expression "-" term
term ::= factor | term "*" factor | term "%" factor
factor :: identifier | number_const | bool_const | "(" expression ")"
count < k (count-identifier; < -relational_op; identifier
stmtlist ::= stmt | stmt ";" stmtlist
stmt ::= assignment | iostmt | ifstmt | whilestmt
assignment ::= identifier ":=" expression ";"
isPrime:=true; (isPrime-identifier, true-factor (bool_const))
whilestmt ::= "WHILE" "(" condition ")" "begin" stmtlist "end"
condition ::= expression relational_op expression
expression ::=term | expresion "+" term | expression "-" term
term ::= factor | term "*" factor | term "%" factor
factor :: identifier | number_const | bool_const | "(" expression ")"
i*i<=num(identifier*identifier <= identifier)
stmtlist ::= stmt | stmt ";" stmtlist
stmt ::= assignment | iostmt | ifstmt | whilestmt
ifstmt ::= "IF" "(" condition ")" "begin" stmtlist "end"
(num % i = 0) expression relational_op expression
num%i term | expresion "+" term | expression "-" term
term ::= factor | term "*" factor | term "%" factor
term "%" factor
identifier “%” identifier
(num % i = 0) expression relational_op number_const