```
START                       //program ::= “START" decllist ";" cmpdstmt "END"
const k;                    // declaration ::= type identifier ";"
                            //type ::="int" |"bool"|"var"|"const"|"let"
                            //identifier ::= letter | letter  {letter | digit}
                            //letter ::= "a" | "b" | "c" | ... | "z"
                            // digit ::= "0" | "1" | ... | "9"
let i;                       // declaration ::= type identifier ";"
let j;                       // declaration ::= type identifier ";"
bool isPrime;               // declaration ::= type identifier ";"
read(k);                    // iostmt ::= "read" | "write" "(" expression ")"
for ( i=2; i < k; i++ ){     //forstmt ::= "for" "(" condition ")" "{" stmtlist "}"
                            //condition ::= expression relation expression
                            //expression ::= expresion "+" term | term
                            //term ::= factor | term "*" factor | term "%" factor
                            // factor :: identifier | number_const | bool_const | "(" expression ")"
isPrime=”true”;             // assignment ::= identifier "=" expression ";"
for (  j=3; j< k*k; j++ ){   //    forstmt ::= "for" "(" condition ")" "{" stmtlist "}"
        if ( i%j===0 ) then
//ifstmt ::= "if" "(" condition ")" "then" ["{"] stmtlist ["}"] [“else” ["{"] stmtlist ["}"] ]
//stmtlist ::= stmt | stmt ";" stmtlist
//   stmt ::= assignment | iostmt | ifstmt | forstmt
// condition ::= expression relation expression
//relation ::= "=" | "!=" | "<" | "<=" | ">" | ">=" | "==" | "==="
//expression ::= expresion "+" term | term
                isPrime=”false”;          // assignment ::= identifier "=" expression ";"
}
if (isPrime===”true”) then
//ifstmt ::= "if" "(" condition ")" "then" ["{"] stmtlist ["}"] [“else” ["{"] stmtlist ["}"] ]
        write(i);           // iostmt ::= "read" | "write" "(" expression ")"
}
END                         //program ::= “START" decllist ";" cmpdstmt "END"
```