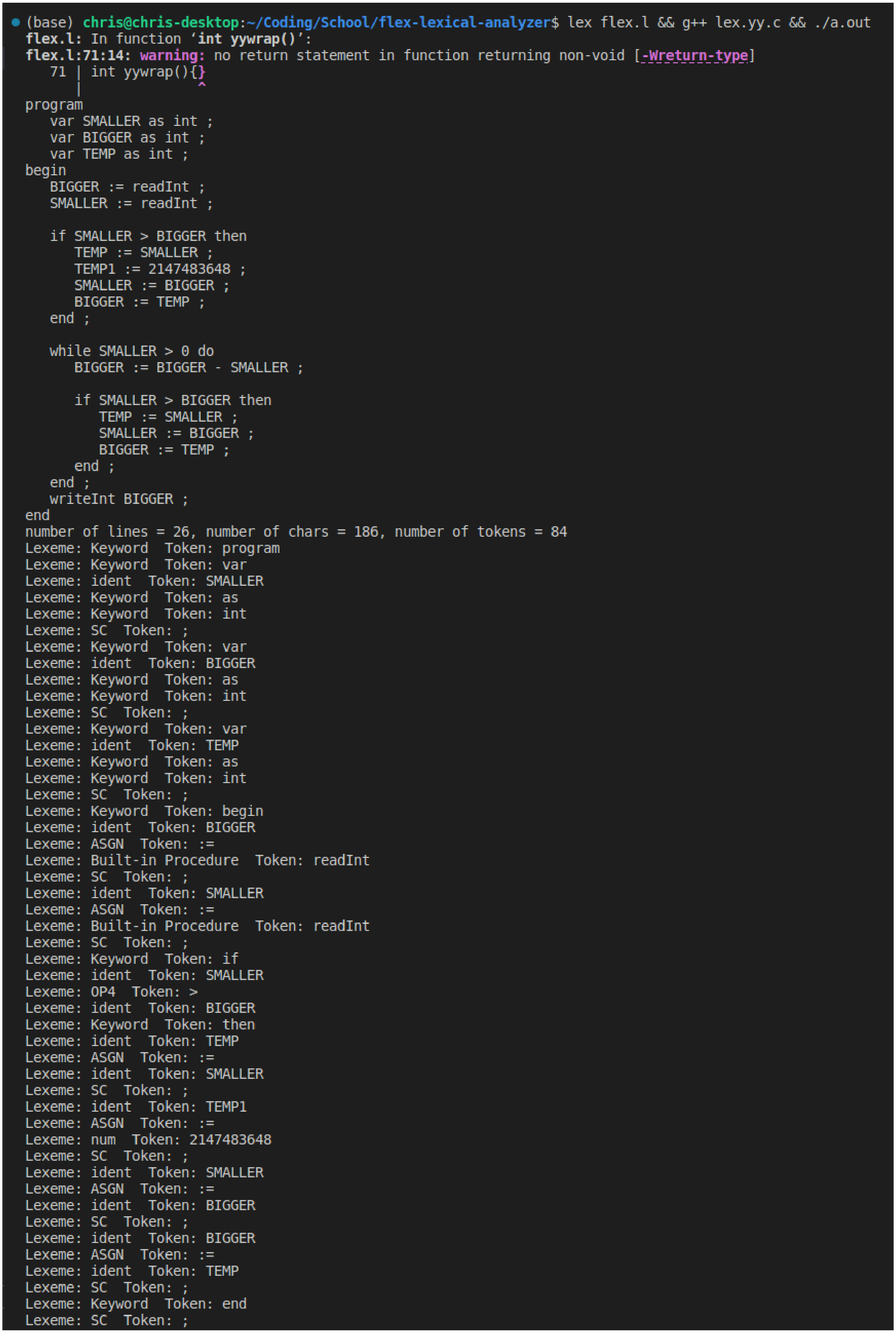
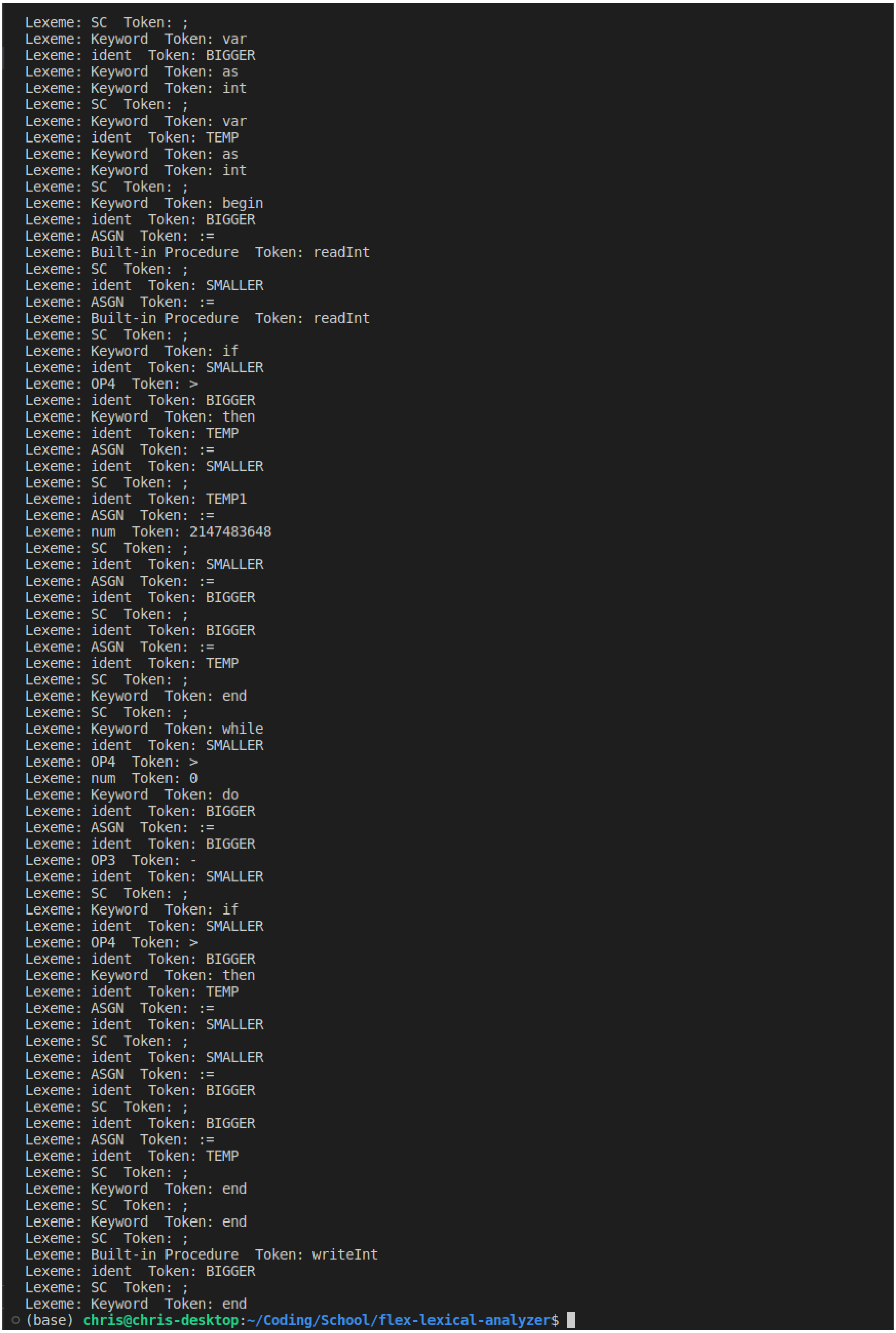


```
● (base) chris@chris-desktop:~/Coding/School/flex-lexical-analyzer$ lex flex.l && g++ lex.yy.c && ./a.out
  flex.l: In function ‘int yywrap()’:
  flex.l:71:14: warning: no return statement in function returning non-void [-Wreturn-type]
     71 | int yywrap(){}
        |              ^
  program
     var SMALLER as int ;
     var BIGGER as int ;
     var TEMP as int ;
  begin
     BIGGER := readInt ;
     SMALLER := readInt ;

     if SMALLER > BIGGER then
        TEMP := SMALLER ;
        TEMP1 := 2147483648 ;
        SMALLER := BIGGER ;
        BIGGER := TEMP ;
     end ;

     while SMALLER > 0 do
        BIGGER := BIGGER - SMALLER ;

        if SMALLER > BIGGER then
           TEMP := SMALLER ;
           SMALLER := BIGGER ;
           BIGGER := TEMP ;
        end ;
     end ;
     writeInt BIGGER ;
  end
  number of lines = 26, number of chars = 186, number of tokens = 84
  Lexeme: Keyword  Token: program
  Lexeme: Keyword  Token: var
  Lexeme: ident  Token: SMALLER
  Lexeme: Keyword  Token: as
  Lexeme: Keyword  Token: int
  Lexeme: SC  Token: ;
  Lexeme: Keyword  Token: var
  Lexeme: ident  Token: BIGGER
  Lexeme: Keyword  Token: as
  Lexeme: Keyword  Token: int
  Lexeme: SC  Token: ;
  Lexeme: Keyword  Token: var
  Lexeme: ident  Token: TEMP
  Lexeme: Keyword  Token: as
  Lexeme: Keyword  Token: int
  Lexeme: SC  Token: ;
  Lexeme: Keyword  Token: begin
  Lexeme: ident  Token: BIGGER
  Lexeme: ASGN  Token: :=
  Lexeme: Built-in Procedure  Token: readInt
  Lexeme: SC  Token: ;
  Lexeme: ident  Token: SMALLER
  Lexeme: ASGN  Token: :=
  Lexeme: Built-in Procedure  Token: readInt
  Lexeme: SC  Token: ;
  Lexeme: Keyword  Token: if
  Lexeme: ident  Token: SMALLER
  Lexeme: OP4  Token: >
  Lexeme: ident  Token: BIGGER
  Lexeme: Keyword  Token: then
  Lexeme: ident  Token: TEMP
  Lexeme: ASGN  Token: :=
  Lexeme: ident  Token: SMALLER
  Lexeme: SC  Token: ;
  Lexeme: ident  Token: TEMP1
  Lexeme: ASGN  Token: :=
  Lexeme: num  Token: 2147483648
  Lexeme: SC  Token: ;
  Lexeme: ident  Token: SMALLER
  Lexeme: ASGN  Token: :=
  Lexeme: ident  Token: BIGGER
  Lexeme: SC  Token: ;
  Lexeme: ident  Token: BIGGER
  Lexeme: ASGN  Token: :=
  Lexeme: ident  Token: TEMP
  Lexeme: SC  Token: ;
  Lexeme: Keyword  Token: end
  Lexeme: SC  Token: ;
```



```
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: var
Lexeme: ident  Token: BIGGER
Lexeme: Keyword  Token: as
Lexeme: Keyword  Token: int
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: var
Lexeme: ident  Token: TEMP
Lexeme: Keyword  Token: as
Lexeme: Keyword  Token: int
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: begin
Lexeme: ident  Token: BIGGER
Lexeme: ASGN  Token: :=
Lexeme: Built-in Procedure  Token: readInt
Lexeme: SC  Token: ;
Lexeme: ident  Token: SMALLER
Lexeme: ASGN  Token: :=
Lexeme: Built-in Procedure  Token: readInt
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: if
Lexeme: ident  Token: SMALLER
Lexeme: OP4  Token: >
Lexeme: ident  Token: BIGGER
Lexeme: Keyword  Token: then
Lexeme: ident  Token: TEMP
Lexeme: ASGN  Token: :=
Lexeme: ident  Token: SMALLER
Lexeme: SC  Token: ;
Lexeme: ident  Token: TEMP1
Lexeme: ASGN  Token: :=
Lexeme: num  Token: 2147483648
Lexeme: SC  Token: ;
Lexeme: ident  Token: SMALLER
Lexeme: ASGN  Token: :=
Lexeme: ident  Token: BIGGER
Lexeme: SC  Token: ;
Lexeme: ident  Token: BIGGER
Lexeme: ASGN  Token: :=
Lexeme: ident  Token: TEMP
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: end
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: while
Lexeme: ident  Token: SMALLER
Lexeme: OP4  Token: >
Lexeme: num  Token: 0
Lexeme: Keyword  Token: do
Lexeme: ident  Token: BIGGER
Lexeme: ASGN  Token: :=
Lexeme: ident  Token: BIGGER
Lexeme: OP3  Token: -
Lexeme: ident  Token: SMALLER
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: if
Lexeme: ident  Token: SMALLER
Lexeme: OP4  Token: >
Lexeme: ident  Token: BIGGER
Lexeme: Keyword  Token: then
Lexeme: ident  Token: TEMP
Lexeme: ASGN  Token: :=
Lexeme: ident  Token: SMALLER
Lexeme: SC  Token: ;
Lexeme: ident  Token: SMALLER
Lexeme: ASGN  Token: :=
Lexeme: ident  Token: BIGGER
Lexeme: SC  Token: ;
Lexeme: ident  Token: BIGGER
Lexeme: ASGN  Token: :=
Lexeme: ident  Token: TEMP
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: end
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: end
Lexeme: SC  Token: ;
Lexeme: Built-in Procedure  Token: writeInt
Lexeme: ident  Token: BIGGER
Lexeme: SC  Token: ;
Lexeme: Keyword  Token: end
○ (base) chris@chris-desktop:~/Coding/School/flex-lexical-analyzer$ 
```